Switch Security Issues: Mitigating VLAN, Spoof, and STP Attacks

Part 2: VLANs Attacks

Cabrillo College

CIS 187 Multilayer Switched Networks

CCNP 3 version 4

Rick Graziani

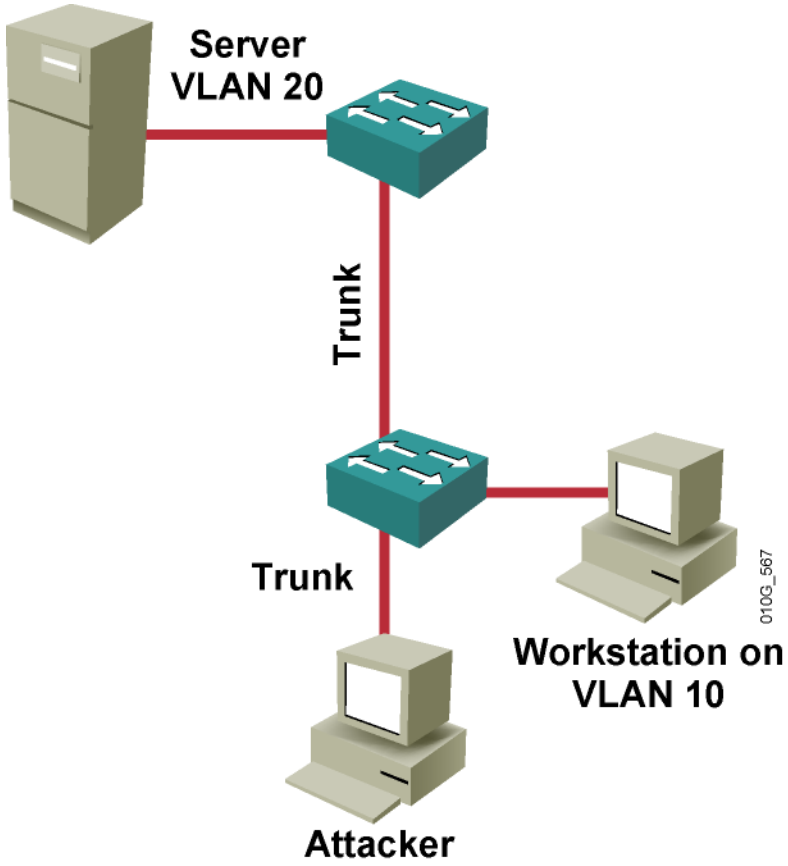
Fall 2006

Switch Attack Categories

- MAC layer attacks
- **VLAN attacks**
- Spoofing attacks
- Attacks on switch devices

VLAN Hopping Attacks

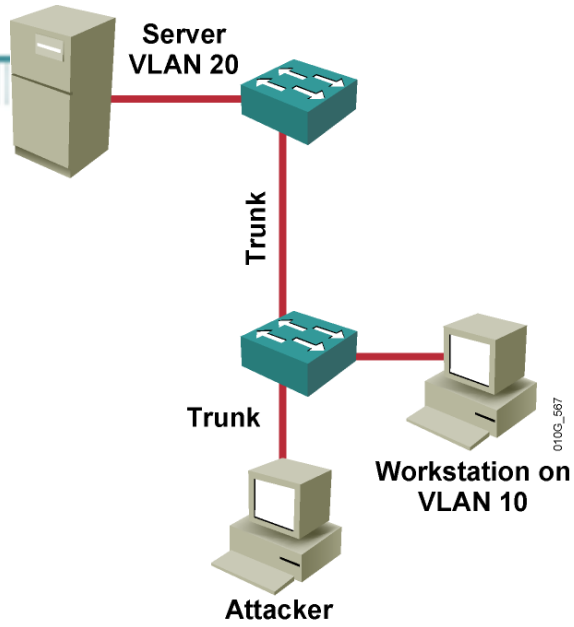
- On networks using trunking protocols, there is a possibility of rogue traffic "***hopping***" from one VLAN to another, thereby creating security vulnerabilities.
- These VLAN Hopping attacks are best mitigated by close control of trunk links:
  - VLAN Access Control Lists (VACLs)
  - Private VLANs (PVLANs).

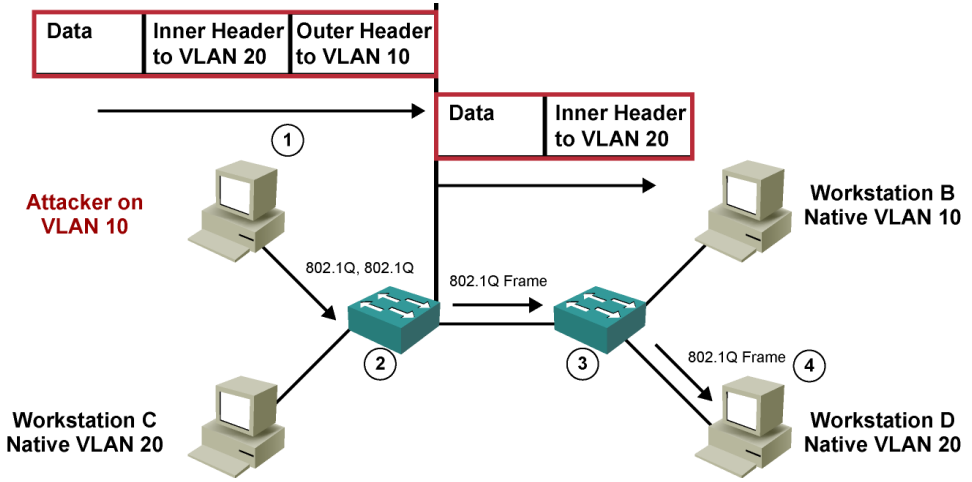


Explaining VLAN Hopping

- **VLAN hopping attack** where an end system sends packets to, or collects packets from, a VLAN that should not be accessible to that end system.
- This is done by:
  - **Switch spoofing**: By negotiating a trunk link in order to send or receive traffic on penetrated VLANs

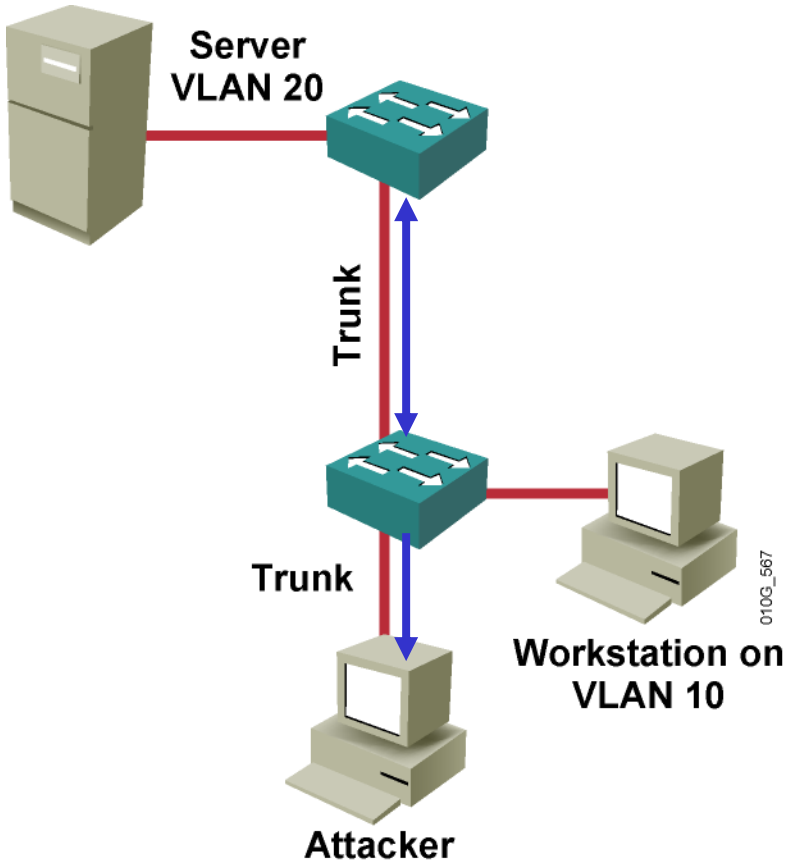
  or
  - **Double tagging**: Tagging the invasive traffic with a specific VLAN ID





VLAN Hopping: Switch Spoofing

- **Switch spoofing attack**, the network attacker configures a system to spoof itself as a switch by emulating Inter-Switch Link (ISL) or 802.1Q signaling along with Dynamic Trunking Protocol (DTP) signaling an attempt to establish a trunk connection to the switch.
- Attacking system spoofs itself as a legitimate trunk negotiating device.
- Trunk link is negotiated dynamically.
- Attacking device gains access to data on all VLANs carried by the negotiated trunk.



“I’m a switch”

VLAN Hopping: DTP

	Dynamic Auto	Dynamic Desirable	Trunk	Access
Dynamic Auto	Access	Trunk	Trunk	Access
Dynamic Desirable	Trunk	Trunk	Trunk	Access
Trunk	Trunk	Trunk	Trunk	Not recommended
Access	Access	Access	Not recommended	Access

Note: Table assumes DTP is enabled at both ends.

- **`show dtp interface` – to determine current setting**

VLAN Hopping: switchport mode access


```
Switch(config)#interface range fa 0/11 - 15
Switch(config-if-range)#switchport mode access
Switch(config-if-range)#switchport access vlan 10

Switch(config)#interface range fa 0/16 - 17
Switch(config-if-range)#switchport mode access
Switch(config-if-range)#switchport access vlan 20
```

- Both of these commands "*should*" be used for access ports:
  - **switchport mode access**
  - **switchport access vlan n**

VLAN Hopping: no switchport mode access


```
Switch(config)#interface range fa 0/11 - 15
Switch(config-if-range)#switchport access vlan 10
Switch(config-if-range)#end

Switch#show interface fa 0/11 switchport
Name: Fa0/11
Switchport: Enabled
Administrative Mode: dynamic desirable
Operational Mode: down
Administrative Trunking Encapsulation: dot1q
Negotiation of Trunking: On
Access Mode VLAN: 10 (Accounting)
Trunking Native Mode VLAN: 1 (default)
Voice VLAN: none
```

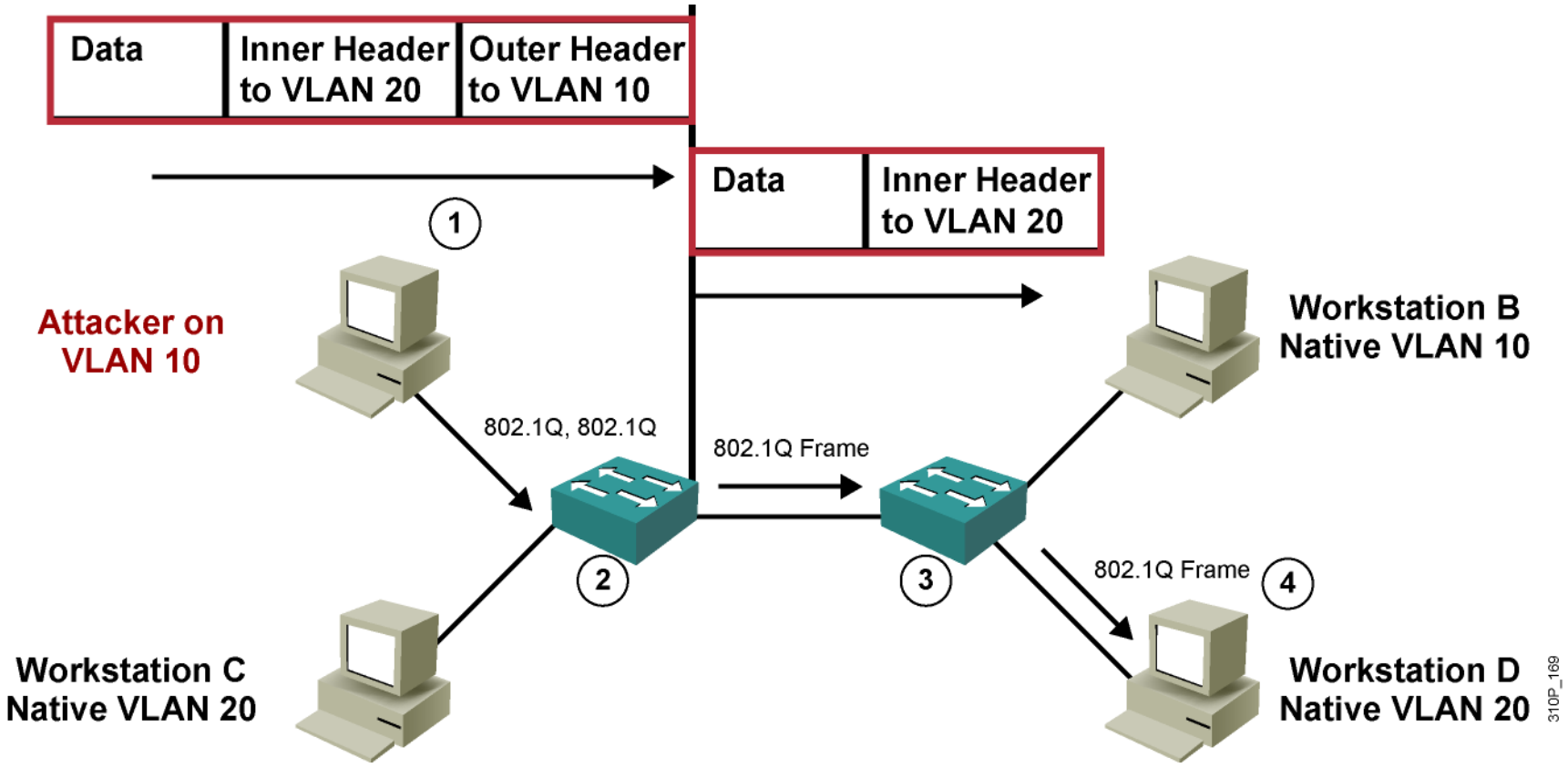
- Without the switchport mode access command, this interface will still try to negotiate trunking.

VLAN Hopping: switchport mode access

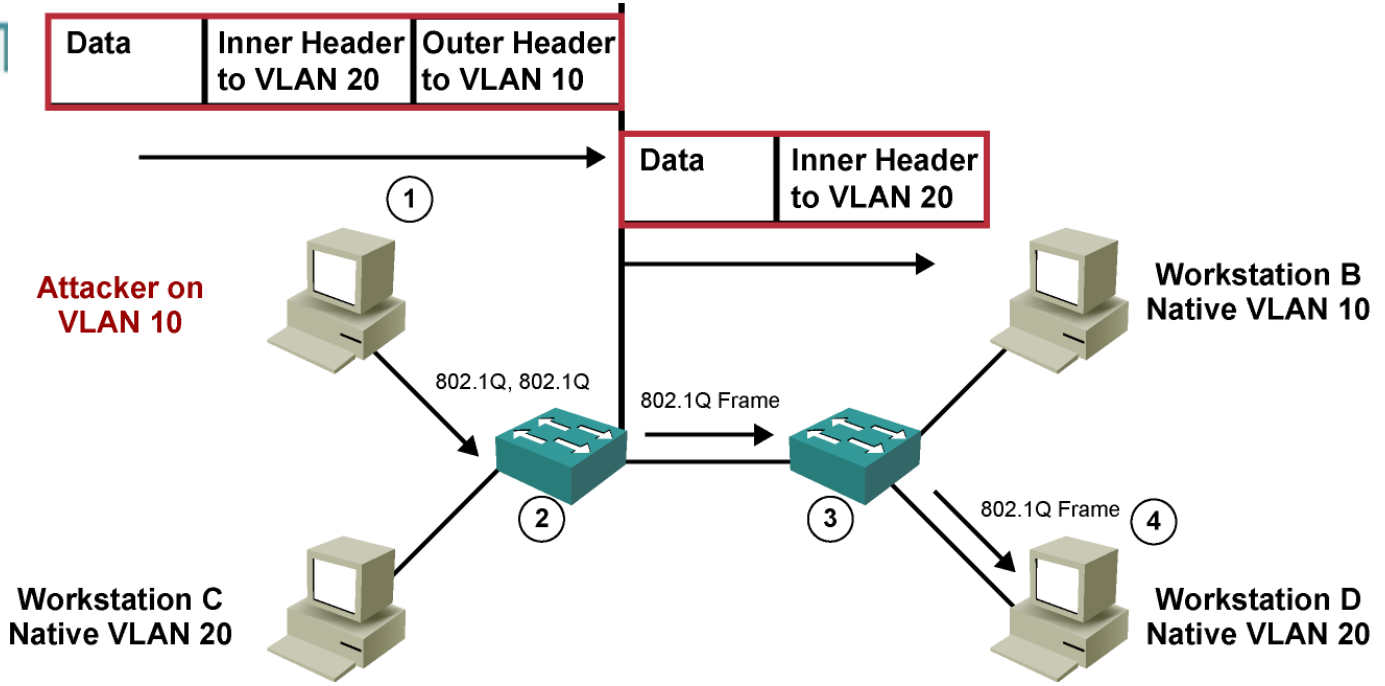

```
Switch(config)#interface range fa 0/11 - 15
Switch(config-if-range)#switchport mode access

Switch#show interface fa 0/11 switchport
Name: Fa0/11
Switchport: Enabled
Administrative Mode: static access
Operational Mode: down
Administrative Trunking Encapsulation: dot1q
Negotiation of Trunking: Off
Access Mode VLAN: 10 (Accounting)
```

- Now configure the range of interfaces for **permanent nontrunking, access mode**
- Notice that negotiation of trunking has been turned off and that this port will **only be a non-trunking access port**.

VLAN Hopping with Double Tagging

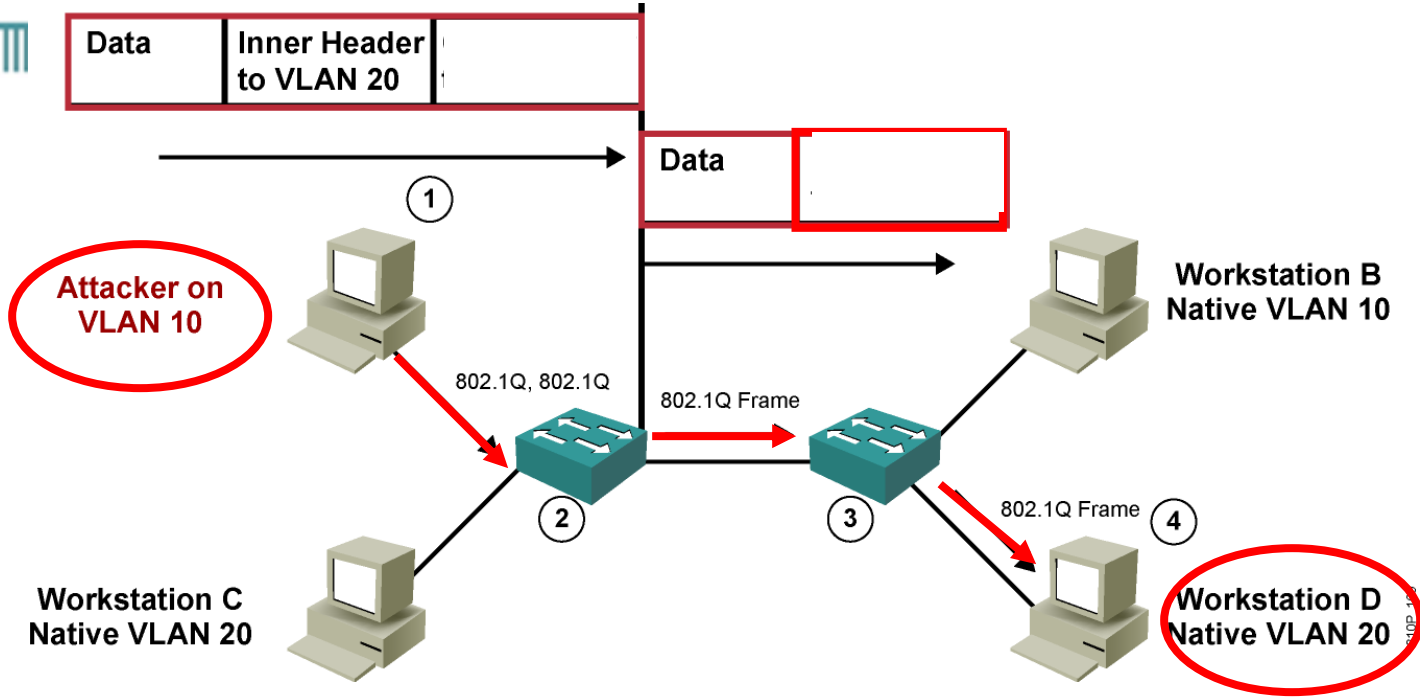


VLAN Hopping with Double Tagging



- Double tagging allows a frame to be forwarded to a destination VLAN other than the source's VLAN.
- Double Tagging Attack is where an attacker workstation generates frames with two 802.1Q headers in order to get the switch to **forward the frames onto a VLAN that would be inaccessible to the attacker through legitimate means**.

VLAN Hopping with Double Tagging



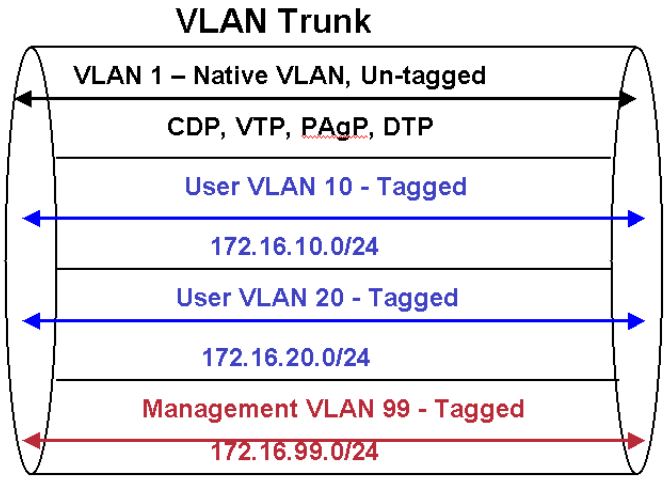
- The first switch to encounter the double-tagged frame strips the first tag off the frame because the **first tag (VLAN 10)** matches the trunk port native VLAN and then forwards the frame out.
- The result is that the **frame is forwarded with the inner 802.1Q tag** out all the switch ports including trunk ports configured with the native VLAN of the network attacker.
- The second switch then forwards the packet to the destination based on the VLAN identifier in the second 802.1Q header.
- Should the trunk not match the native VLAN of the attacker, the frame would be untagged and flooded only to the original VLAN.

Mitigating VLAN Hopping: Access Ports


```
Switch(config)#interface range fa 0/11 - 15
Switch(config-if-range)#switchport mode access
Switch(config-if-range)#switchport access vlan 10

Switch(config)#interface range fa 0/16 - 17
Switch(config-if-range)#shutdown
Switch(config-if-range)#switchport mode access
Switch(config-if-range)#switchport access vlan 999
```

- The measures to defend the network from VLAN hopping are a series of **best practices** for all switch ports and parameters to follow when establishing a trunk port.
- **Access Ports**
  - **Configure all unused ports as access ports** so that trunking cannot be negotiated across those links.
  - Place all **unused ports in the shutdown state** and associate with a VLAN designed only for unused ports, carrying no user data traffic.

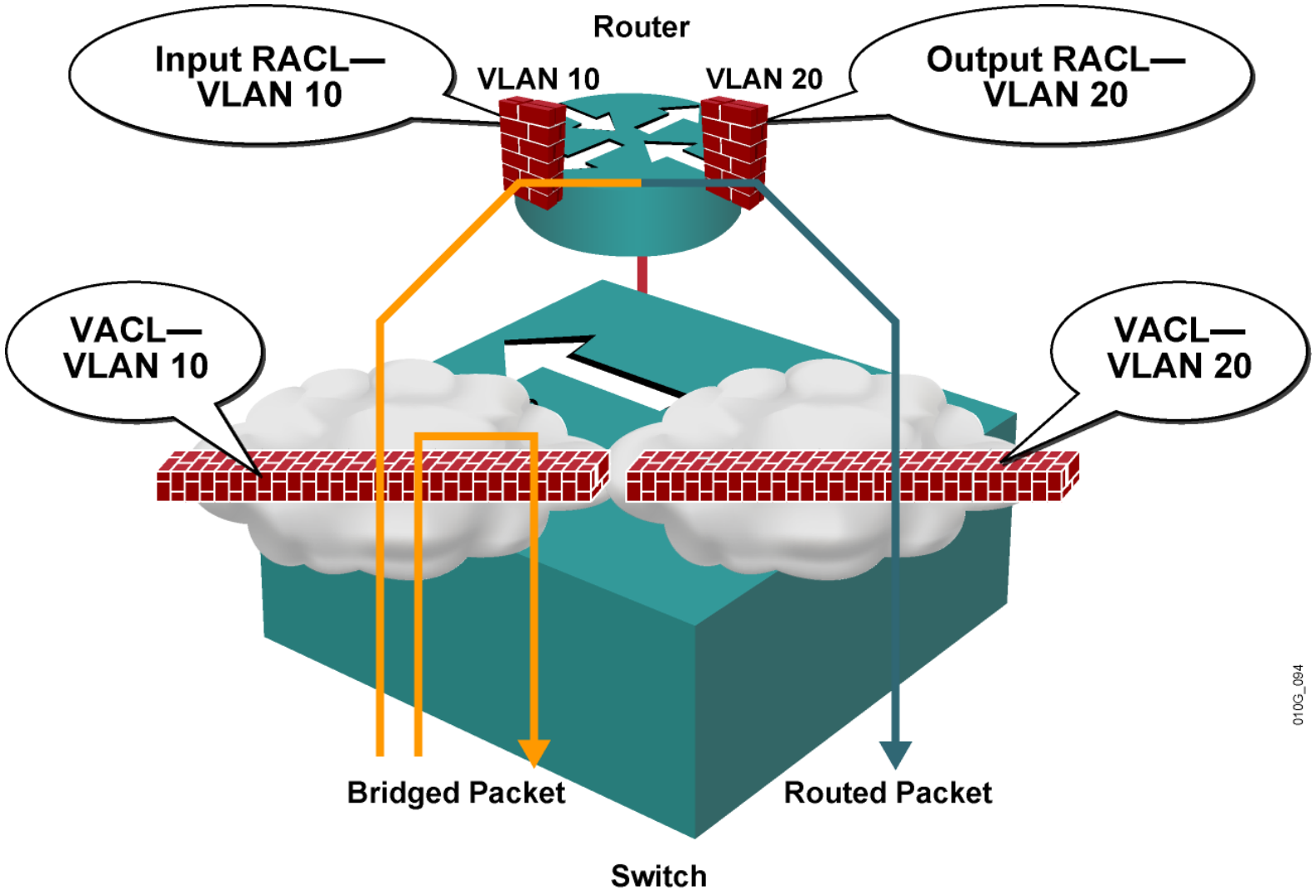
Mitigating VLAN Hopping: Trunk Ports




```
Switch(config)#interface gig 0/1
Switch(config-if-range)#switchport mode trunk
Switch(config-if-range)#switchport trunk native vlan 1
Switch(config-if-range)#switchport trunk allowed vlan 1,10,20,99
```

- **Trunk Ports**
  - When establishing a trunk link, purposefully configure the following:
    - The **native VLAN to be different from any data VLANs** (VLAN 1 is the default)
    - **Trunking as "on,"** rather than negotiated
    - The **specific VLAN range to be carried on the trunk**
- Note: The configuration commands in the figure will not work on access ports that support VoIP because they will be configured as trunk ports. However, on all other access ports it is best practice to apply these commands to mitigate VLAN hopping.

Types of ACLs



VACLs

- **VACLs** (a.k.a. **VLAN access maps**) apply to all traffic on the VLAN.
- VACLs apply to:
  - **IP traffic**
  - **MAC-Layer traffic**
- **VACLs follow route-map conventions**, in which map sequences are checked in order.
- When a matching permit ACE (Access Control Entry) is encountered, the switch takes the action.
- When a matching deny ACE is encountered, the switch checks the next ACL in the sequence or checks the next sequence.
- Three VACL actions are permitted:
  - **Permit** (with capture, Catalyst 6500 only)
  - **Redirect** (Catalyst 6500 only)
  - **Deny** (with logging, Catalyst 6500 only)

VACLs

Define a VLAN access map.

```
Switch(config)#vlan access-map map_name [seq#]
```

Configure a match clause.

```
Switch(config-access-map)# match {ip address {1-199 |
1300-2699 | acl_name} | ipx address {800-999 | acl_name}|
mac address acl name}
```

- You can use the optional ***sequence-number*** to indicate the position a new route map is to have in the list of route maps already configured with the same name.
- If you don't specify a sequence number, the first route map condition will be automatically numbered as ***10***.

VACLs

Define a VLAN access map.

```
Switch(config)#vlan access-map map_name [seq#]
```

Configure a match clause.

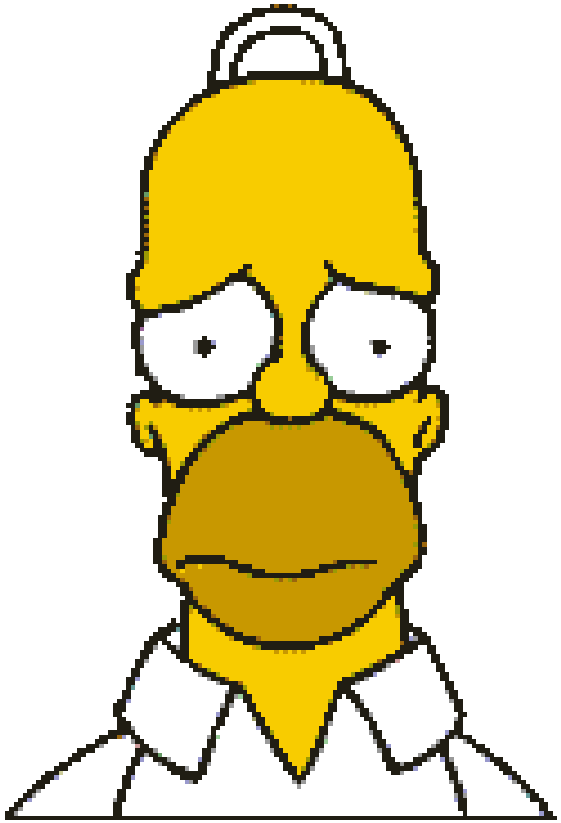
```
Switch(config-access-map)# match {ip address {1-199 |
1300-2699 | acl_name} | ipx address {800-999 | acl_name}|
mac address acl name}
```

Once you have entered the **vlan map** command, you can enter **set** and **match** commands in the route-map configuration mode.

- Each **route-map** command has a list of **match** and **action** commands associated with it.
- The **match** commands specify the *match criteria*—the conditions that should be tested to determine whether or not to take action.
- The **action** commands specify the *actions*—the actions to perform if the match criteria are met.

VACLs

Don’t worry, several examples will help show how this works…



VLAN Map Configuration Guidelines

- If there is **no router ACL** configured to deny traffic on a routed VLAN interface (input or output), and ***no* VLAN map configured**, **all traffic is permitted.**
- Each VLAN map consists of a series of entries.
  - The **order** of entries in an VLAN map is **important**.
  - A **packet** that comes into the switch is **tested against the first entry** in the VLAN map.
  - If it **matches**, the **action** specified for that part of the VLAN map is **taken**.
  - If there is **no match**, the packet is **tested against the next entry** in the map.
- If the VLAN map has at least ***one match clause for the type of packet*** (IP or MAC) and the **packet does not match any of these match clauses**, the **default is to drop the packet**.
- If there is ***no match clause for that type of packet*** in the VLAN map, the default is to **forward the packet**.

Configuring VACLs

To configure VACLs, complete the following steps.

Step	Description
1.	Define a VLAN access map. `Switch(config)#vlan access-map map_name [seq#]`
2.	Configure a match clause. `Switch(config-access-map)# match {ip address {1-199 | 1300-2699 | acl_name} | ipx address {800-999 | acl_name}| mac address acl name}`
3.	Configure an action clause. Switch(config-access-map)#action {drop [log]} | {forward [capture]} | {redirect {{**fastethernet** | **gigabitethernet** | **tengigabitethernet**} *slot/port*} | {**port-channel** *channel_id*}}
4.	Apply a map to VLANs. `Switch(config)#vlan filter map_name vlan_list list`

5.	Verify the VACL configuration.
	`Switch#show vlan access-map map_name`
	`Switch#show vlan filter [ access-map map_name | vlan_id ]`

Example 1

- Drop packets with source IP 10.1.0.0/16 in VLANs 1-4094.
- (Default) Drop all other IP packets: VLAN map has at least one match clause, ip address
- (Default) Forward all non-IP packets: Forward all other “frames”, no match clauses

3

```
Switch(config)# access-list 1 permit 10.1.0.0 0.0.255.255
```

2

```
Switch(config)# vlan access-map PxR1 10
Switch(config-access-map)# match ip address 1
Switch(config-access-map)# action drop
Switch(config)# vlan access-map PxR1 20
Switch(config-access-map)# action forward
```

1

```
Switch(config)# vlan filter PxR1 vlan_list 1-4094
```

```
vlan access-map PxR1 10
 action drop
 match ip address 1
vlan access-map PxR1 20
 action forward
vlan filter PxR1  vlan-list 1-4094
!
access-list 1 permit 10.1.0.0 0.0.255.255
```

Example 2

- In this example, the VLAN map has a
  - default action of drop for IP packets
  - default action of forward for MAC packets
- Used with standard ACL 101 and extended named access lists **igmp-match** and **tcp-match**, the map will have the following results:
  - Forward all UDP packets
  - Drop all IGMP packets
  - Forward all TCP packets
  - Drop all other IP packets
  - Forward all non-IP packets

- Forward all UDP packets
- Drop all IGMP packets
- Forward all TCP packets
- (Default) Drop all other IP packets: VLAN map has at least one match clause, tcp-match
- (Default) Forward all non-IP packets: Forward all other “frames”, no match clauses

```
Switch(config)# access-list 101 permit udp any any

Switch(config)# ip access-list extended igmp-match
Switch(config-ext-nacl)# permit igmp any any

Switch(config)# ip access-list extended tcp-match
Switch(config-ext-nacl)# permit tcp any any
```

ACLs

2

```
Switch(config)# vlan access-map drop-ip-default 10
Switch(config-access-map)# match ip address 101
Switch(config-access-map)# action forward
```

VACLs

3

```
Switch(config)# vlan access-map drop-ip-default 20
Switch(config-access-map)# match ip address igmp-match
Switch(config-access-map)# action drop
```

4

```
Switch(config)# vlan access-map drop-ip-default 30
Switch(config-access-map)# match ip address tcp-match
Switch(config-access-map)# action forward
```

1

```
Switch(config)# vlan filter drop-ip-default vlan-list 10-50
```

Filter

Example 3

- In this example, the VLAN map has a:
  - default action of drop for MAC packets
  - default action of forward for IP packets.
- Used with MAC extended access lists **good-hosts** and **good-protocols**, the map will have the following results:
  - Forward MAC packets from hosts 0000.0c00.0111 and 0000.0c00.0211
  - Forward MAC packets with decnet-iv or vines-ip protocols
  - Drop all other non-IP packets
  - Forward all IP packets

- Forward MAC packets from hosts 0000.0c00.0111 and 0000.0c00.0211
- Forward MAC packets with decnet-iv or vines-ip protocols
- (Default) Drop all other non-IP packets: VLAN map has at least one match clause, good-protocols
- (Default) Forward all IP packets: Forward all other “frames”, no match clauses

```
Switch(config)# mac access-list extended good-hosts
Switch(config-ext-macl)# permit host 000.0c00.0111 any
Switch(config-ext-macl)# permit host 000.0c00.0211 any

Switch(config)# mac access-list extended good-protocols
Switch(config-ext-macl)# permit any any decnet-ip
Switch(config-ext-macl)# permit any any vines-ip
```

MAC ACLs

2

```
Switch(config)# vlan access-map drop-mac-default 10
Switch(config-access-map)# match mac address good-hosts
Switch(config-access-map)# action forward
```

VACLs

3

```
Switch(config)# vlan access-map drop-mac-default 20
Switch(config-access-map)# match mac address good-protocols
Switch(config-access-map)# action forward
```

1

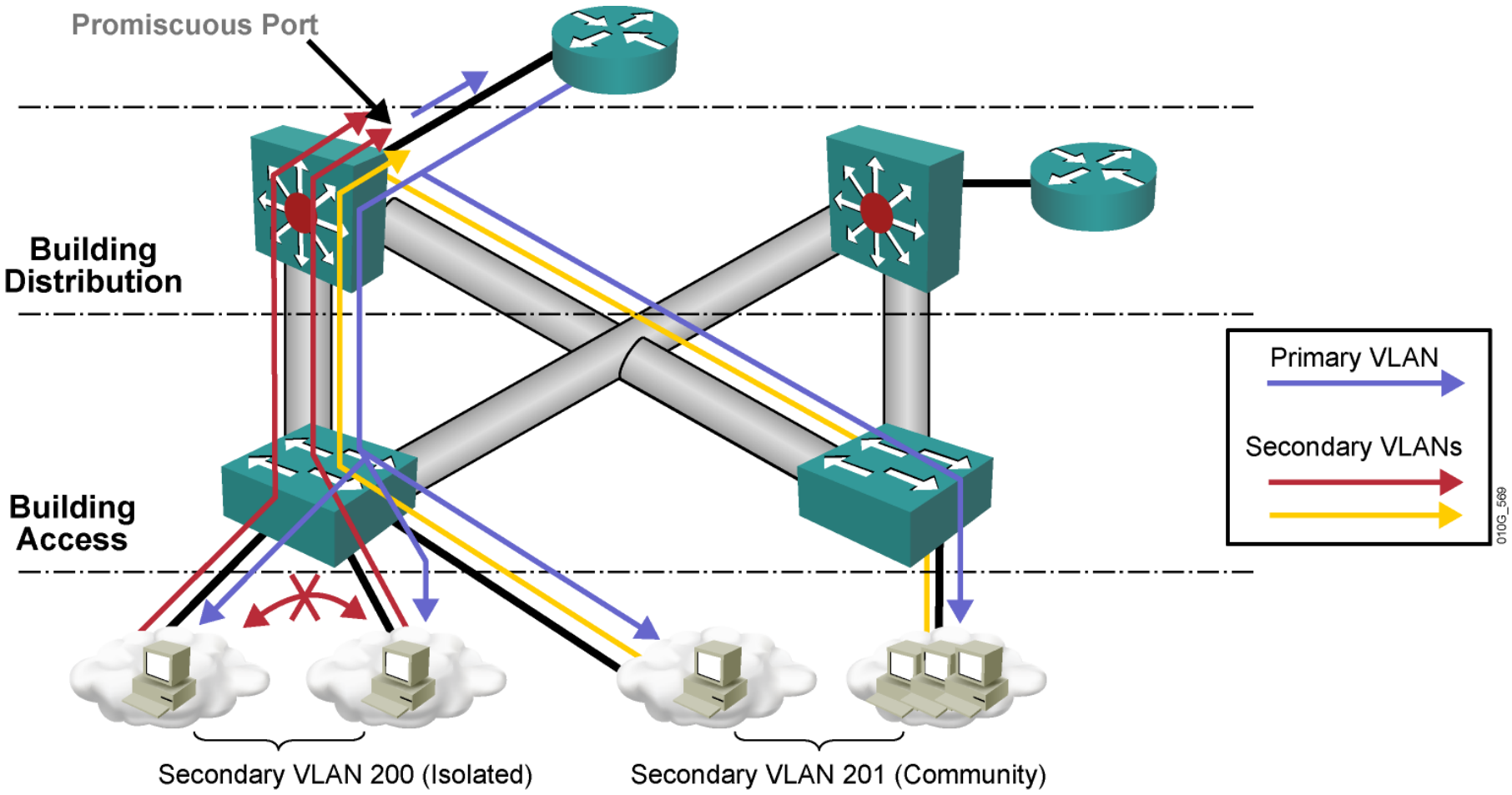
```
Switch(config)# vlan filter drop-mac-default vlan-list 10-50
```

Filter

MAC Extended ACLs

	Command	Purpose
Step 1	**configure terminal**	Enter global configuration mode.
Step 2	**mac access-list extended** *name*	Define an extended MAC access list by using a name.
Step 3	{**deny** | **permit**} {**any** | **host** *source MAC address*} {**any** | **host** *destination MAC address*} [**aarp** | **amber** | **appletalk** | **dec-spanning** | **decnet-iv** | **diagnostic** | **dsm** | **etype-6000** | **etype-8042** | **lat** | **lavc-sca** | **mop-console** | **mop-dump** | **msdos** | **mumps** | **netbios** | **vines-echo** |**vines-ip** | **xns-idp**]	In extended MAC access-list configuration mode, specify to **permit** or **deny any** source MAC address or a specific **host** source MAC address and **any** destination MAC address. (Optional) You can also enter these options: **aarp** | **amber** | **appletalk** | **dec-spanning** | **decnet-iv** | **diagnostic** | **dsm** | **etype-6000** | **etype-8042** | **lat** | **lavc-sca** | **mop-console** | **mop-dump** | **msdos** | **mumps** | **netbios** | **vines-echo** |**vines-ip** | **xns-idp**—(a non-IP protocol).

Private VLANS



Private VLANS

- **Service providers** often have devices from multiple clients, as well as their own servers, on a single Demilitarized Zone (DMZ) segment or VLAN.
  - Catalyst 6500/4500 switches implement private PVLANs to keep some switch ports shared and some switch ports isolated, although all ports exist on the same VLAN.
  - The 2950 and 3550 support “protected ports,” which is functionality similar to PVLANs on a per-switch basis.
- The **traditional solution** to address these Internet service provider (ISP) requirements is to provide **one VLAN per customer**, with each VLAN having its **own IP subnet**.
- Here are the **challenges with this traditional solution**:
  - Supporting a separate VLAN per customer may require a **high number of interfaces** on service provider network devices.
  - **Spanning tree** becomes more complicated with many VLAN iterations.
  - **Network address space must be divided into many subnets**, which wastes space and increases management complexity.
  - **Multiple ACL applications** are required to maintain security on multiple VLANs, resulting in increased management **complexity**.

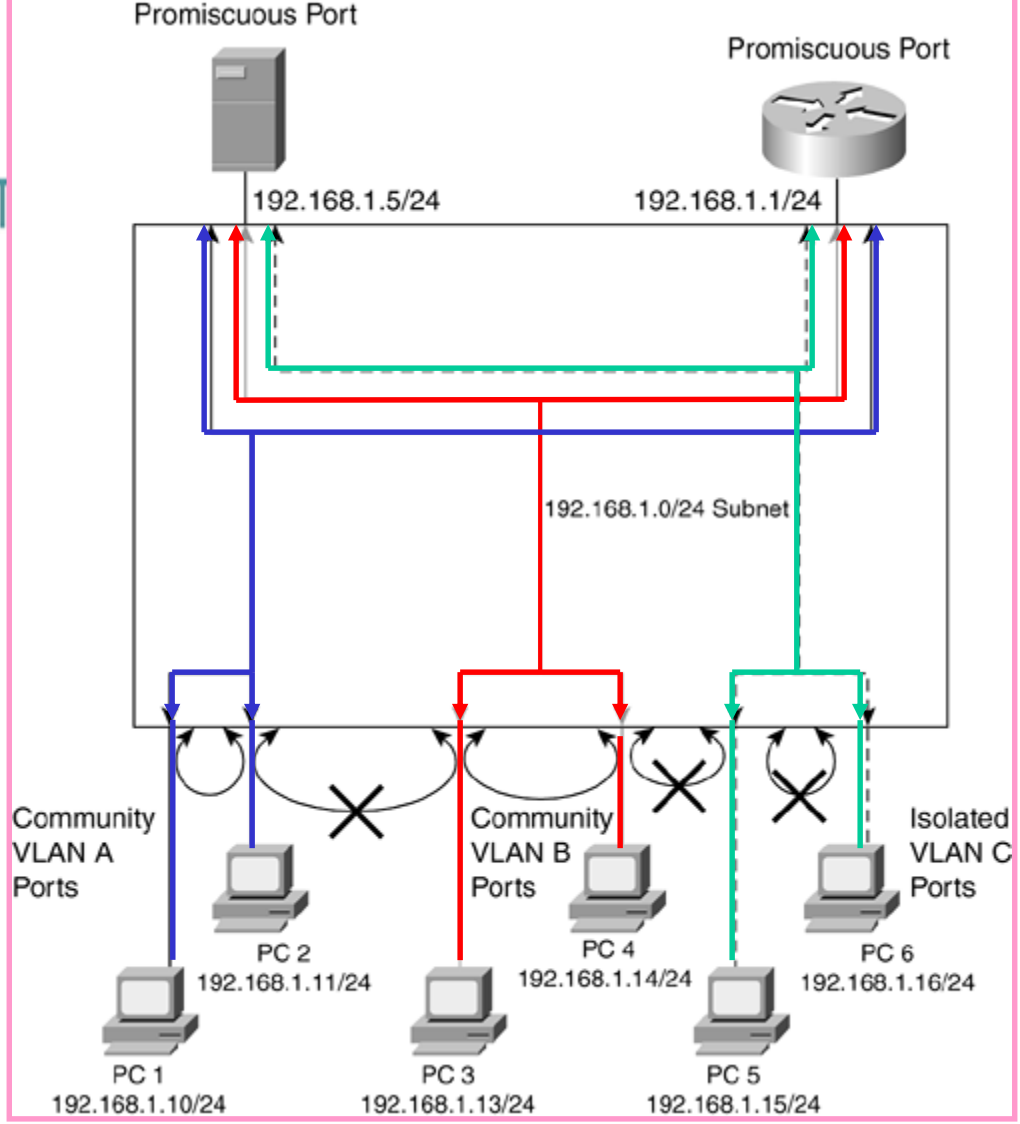
Private VLANS

- **Private VLANs (pVLAN)** are VLANs that provide **isolation between ports within the same VLAN.**
- This isolation eliminates the need for a separate VLAN and IP subnet per customer.
  - Provides **Security**
  - **Reduces the number of IP subnets**
  - **Reduces the VLANs' utilization by isolating traffic** between network devices residing in the same VLAN
- ***Used by Service providers to deploy host services and network access where all devices reside in the same subnet but only communicate to a default gateway, backup servers, or another network.***
- Although network devices reside in different pVLANs, they can use the same IP subnet.
  - Only communicate with the default gateway

Support VLANs

pVLAN consists of two support VLANs

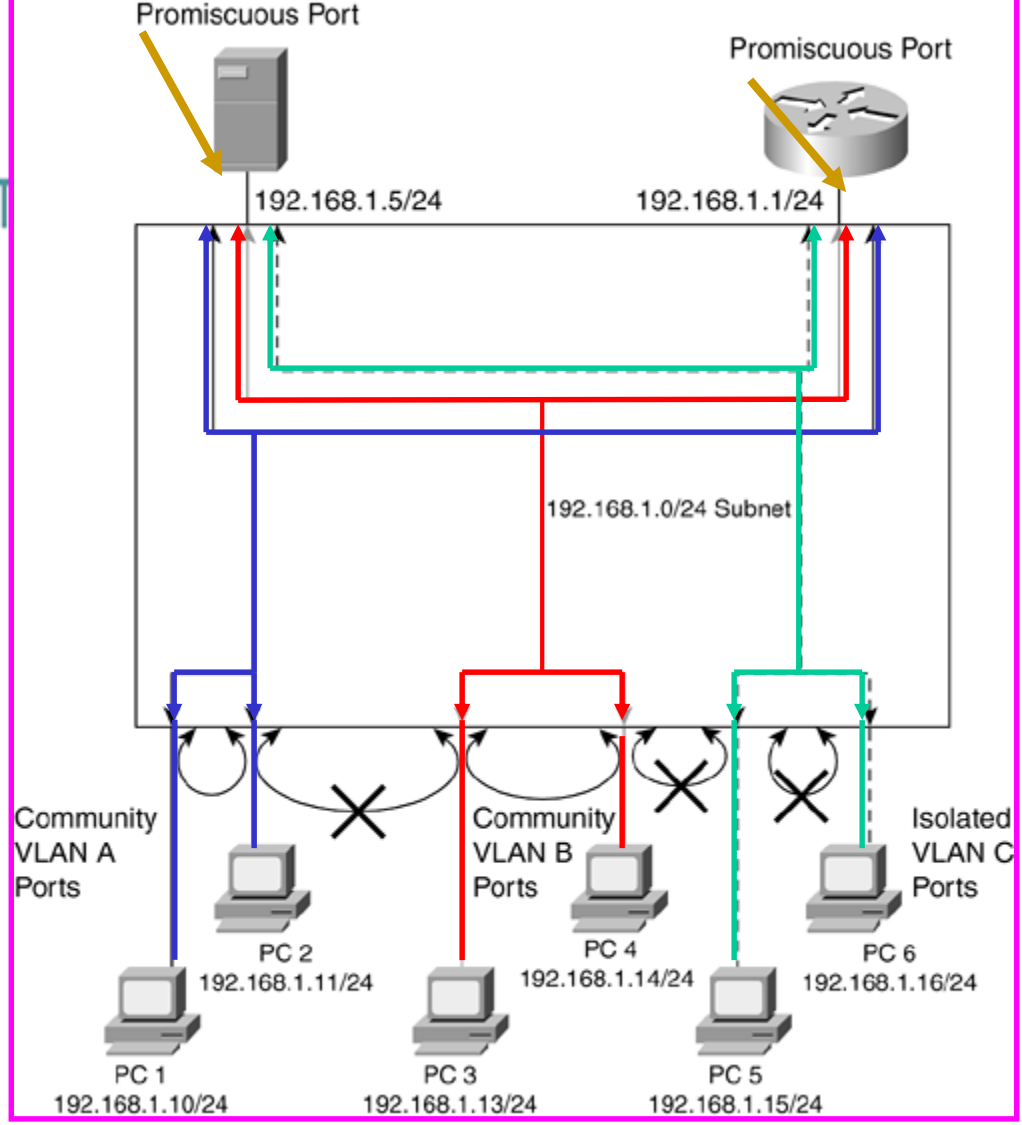
- **Primary VLAN**
  - 192.168.1.0/24 Subnet
  - The high-level VLAN of the pVLAN.
  - Primary VLAN can have many secondary VLANs.
  - Secondary VLANs can have same IP subnet address
- **Secondary VLAN**
  - VLAN A, VLAN B, VLAN C
  - Child to Primary VLAN
  - Mapped to a single Primary VLAN
  - *End devices attach to Secondary VLAN*



Promiscuous Ports

Promiscuous Ports

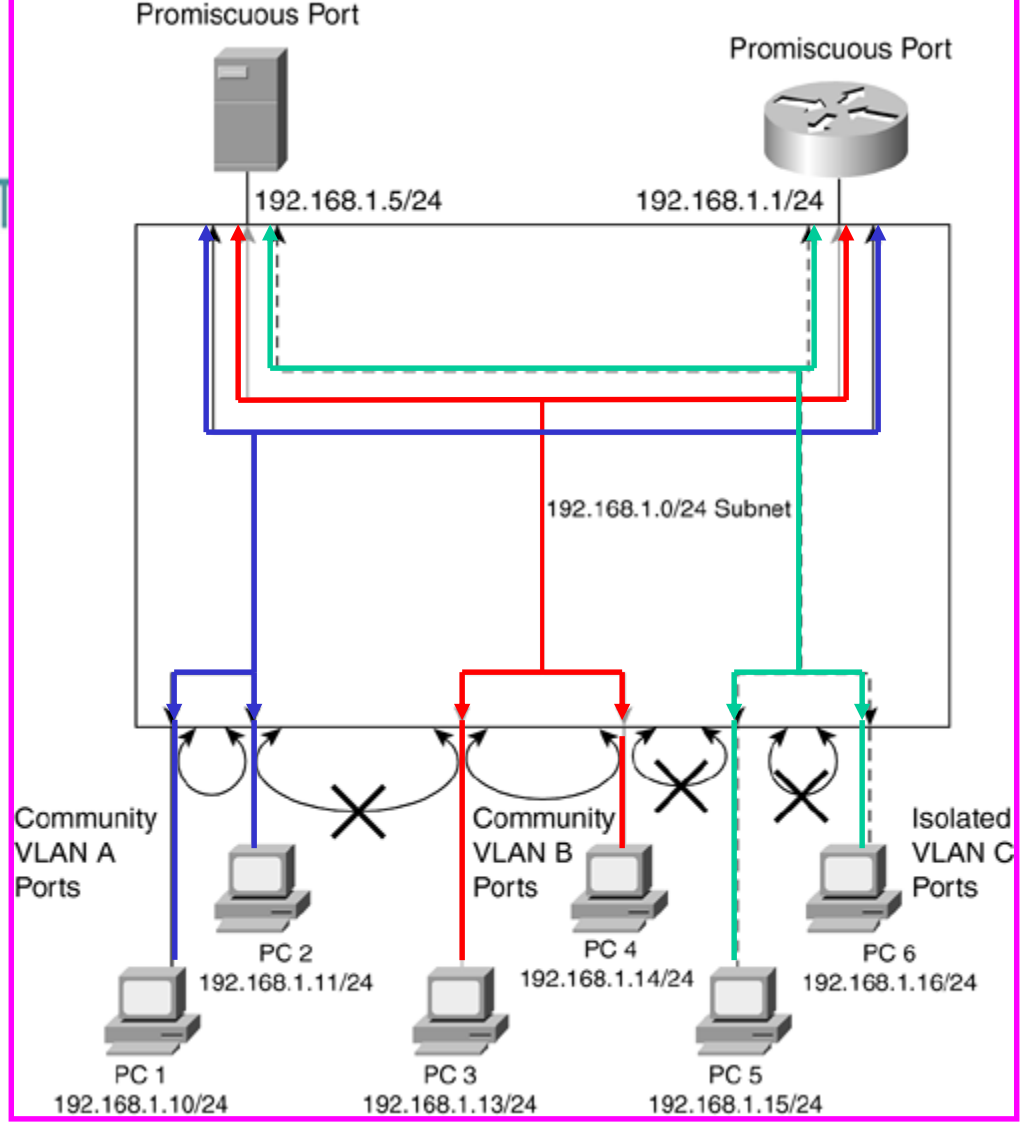
- Promiscuous port is only ***part of one Primary VLAN***
- Promiscuous port can **map to multiple Secondary VLANs**
- Promiscuous port is usually a:
  - Router port
  - Backup server
  - VLAN interface
- All devices in pVLAN communicate with promiscuous ports



Secondary VLANs

Two types of Secondary VLANs

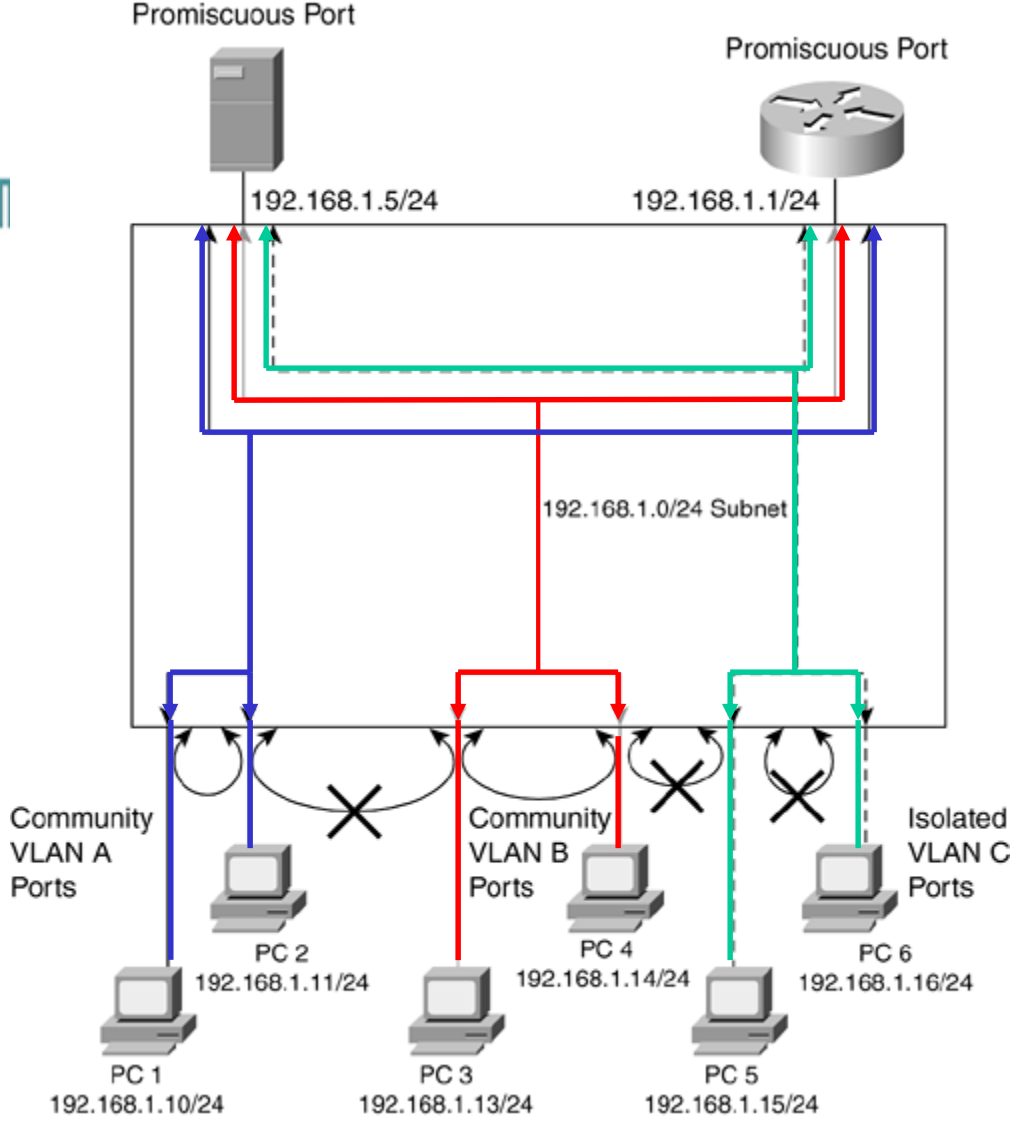
- Community VLANs
  - VLAN A
  - VLAN B
- Isolated VLANs
  - VLAN C



Community VLANs

Community VLANs

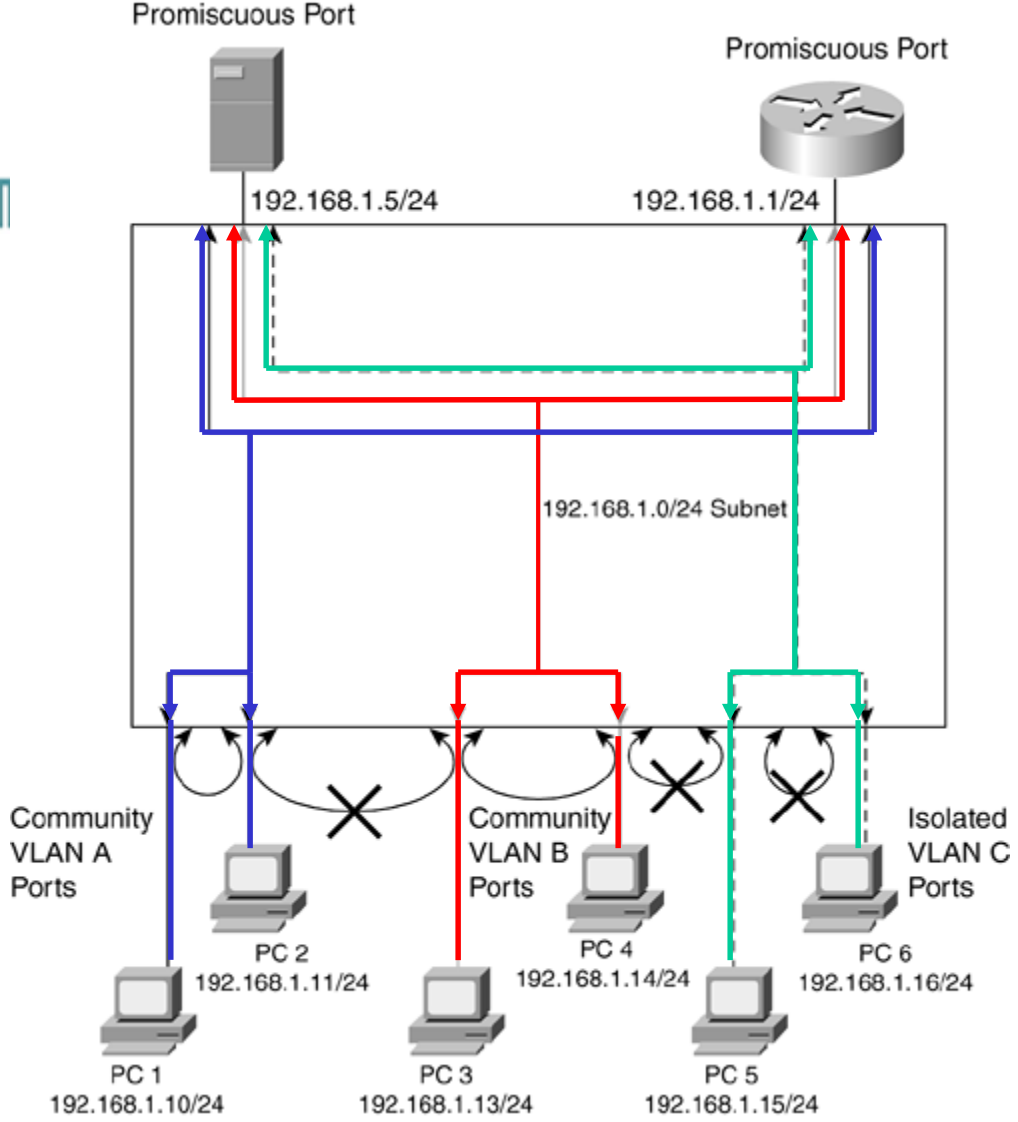
- Can communicate with:
  - Promiscuous ports
  - Other ports in same community
- PC 1 and PC 2 can communicate
- PC 3 and PC 4 can communicate
- PC 2 and PC 3 **cannot** communicate



Isolated VLANs

Isolated VLANs

- Can ***only*** communicate with promiscuous ports
- PC 5 and PC 6 **cannot** communicate
- PC 1 and PC 5 **cannot** communicate
- PC3 and PC 6 **cannot** communicate



Configuring Private VLANs


```
Switch(config-vlan)#private-vlan [primary | isolated |
community]
```

- Configures a VLAN as a private VLAN

```
Switch(config-vlan)#private-vlan association
{secondary_vlan_list | add svl | remove svl}
```

- Associates secondary VLANs with the primary VLAN

```
Switch#show vlan private-vlan type
```

- Verifies private VLAN configuration

Configuring Private VLAN Ports


```
Switch(config-if)#switchport mode private-vlan {host |
promiscuous}
```

- Configures an interface as a private VLAN port

```
Switch(config-if)#switchport private-vlan host-association
{primary_vlan_ID secondary_vlan_ID
```

- Associates an isolated or community port with a private VLAN

```
Switch(config-if)#private-vlan mapping primary_vlan_ID
{secondary_vlan_list | add svl | remove svl}
```

- Maps a promiscuous PVLAN port to a private VLAN

```
Switch#show interfaces private-vlan mapping
```

- Verifies private VLAN port configuration

Configuring PVLANs

To configure a PVLAN, follow these steps.

Step 1 Set VTP mode to transparent.

Step 2 Create the secondary VLANs.

Note Isolated and community VLANs are secondary VLANs.

Step 3 Create the primary VLAN.

Step 4 Associate the secondary VLAN with the primary VLAN. Only one isolated VLAN can be mapped to a primary VLAN, but more than one community VLAN can be mapped to a primary VLAN.

Step 5 Configure an interface as an isolated or community port.

Step 6 Associate the isolated port or community port with the primary-secondary VLAN pair.

Step 7 Configure an interface as a promiscuous port.

Step 8 Map the promiscuous port to the primary-secondary VLAN pair.

Configuring and Associating VLANs in a Private VLAN


```
Switch# configure terminal
Switch(config)# vlan 20
Switch(config-vlan)# private-vlan primary
Switch(config-vlan)# exit

Switch(config)# vlan 501
Switch(config-vlan)# private-vlan isolated
Switch(config-vlan)# exit

Switch(config)# vlan 502
Switch(config-vlan)# private-vlan community
Switch(config-vlan)# exit

Switch(config)# vlan 503
Switch(config-vlan)# private-vlan community
Switch(config-vlan)# exit

Switch(config)# vlan 20
Switch(config-vlan)# private-vlan association 501-503
```

- Create Primary VLAN 20

Configuring and Associating VLANs in a Private VLAN


```
Switch# configure terminal
Switch(config)# vlan 20
Switch(config-vlan)# private-vlan primary
Switch(config-vlan)# exit

Switch(config)# vlan 501
Switch(config-vlan)# private-vlan isolated
Switch(config-vlan)# exit

Switch(config)# vlan 502
Switch(config-vlan)# private-vlan community
Switch(config-vlan)# exit

Switch(config)# vlan 503
Switch(config-vlan)# private-vlan community
Switch(config-vlan)# exit

Switch(config)# vlan 20
Switch(config-vlan)# private-vlan association 501-503
```

- Create two Secondary VLANs
  - Isolated (501)
  - Community (502, 503)

Configuring and Associating VLANs in a Private VLAN


```
Switch# configure terminal
Switch(config)# vlan 20
Switch(config-vlan)# private-vlan primary
Switch(config-vlan)# exit

Switch(config)# vlan 501
Switch(config-vlan)# private-vlan isolated
Switch(config-vlan)# exit

Switch(config)# vlan 502
Switch(config-vlan)# private-vlan community
Switch(config-vlan)# exit

Switch(config)# vlan 503
Switch(config-vlan)# private-vlan community
Switch(config-vlan)# exit

Switch(config)# vlan 20
Switch(config-vlan)# private-vlan association 501-503
```

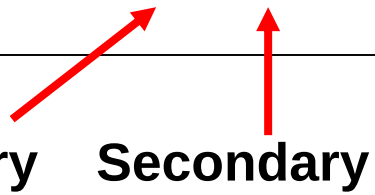
- Associate Secondary VLANs (501-503) with Primary VLAN (20)

Configuring and Associating VLANs in a Private VLAN


```
Switch(config)# show vlan private vlan
Primary Secondary Type Ports
------- --------- -----------------
20 501 isolated
20 502 community
20 503 community
```

Configuring a Layer 2 Interface as a Private-VLAN Host Port


```
Switch# configure terminal
Switch(config)# interface fastethernet0/22
Switch(config-if)# switchport mode private-vlan host
Switch(config-if)# switchport private-vlan host-association 20 25
```



	Command	Purpose
Step 1	**configure terminal**	Enter global configuration mode.
Step 2	**interface** *interface-id*	Enter interface configuration mode for the Layer 2 interface to be configured.
Step 3	**switchport mode private-vlan host**	Configure the Layer 2 port as a private-VLAN host port.
Step 4	**switchport private-vlan host-association** *primary_vlan_id secondary_vlan_id*	Associate the Layer 2 port with a private VLAN.

Configuring a Layer 2 Interface as a Private-VLAN Host Port


```
Switch# show interfaces fastethernet0/22 switchport
Name: Fa0/22
Switchport: Enabled
Administrative Mode: private-vlan host
Operational Mode: private-vlan host
Administrative Trunking Encapsulation: negotiate
Operational Trunking Encapsulation: native
Negotiation of Trunking: Off
Access Mode VLAN: 1 (default)
Trunking Native Mode VLAN: 1 (default)
Administrative Native VLAN tagging: enabled
Voice VLAN: none
Administrative private-vlan host-association: 20 (VLAN0020) 25
   (VLAN0025)
Administrative private-vlan mapping: none
Administrative private-vlan trunk native VLAN: none
Administrative private-vlan trunk Native VLAN tagging: enabled
Administrative private-vlan trunk encapsulation: dot1q
Administrative private-vlan trunk normal VLANs: none
Administrative private-vlan trunk private VLANs: none
Operational private-vlan:
20 (VLAN0020) 25 (VLAN0025)
```

PVLAN Port Types

- Isolated: Communicate only with promiscuous ports
- Promiscuous: Communicate with all other ports
- Community: Communicate with other members of community and all promiscuous ports

Configuring Private VLANs


```
Switch(config-vlan)#private-vlan [primary | isolated |
community]
```

- Configures a VLAN as a private VLAN

```
Switch(config-vlan)#private-vlan association
{secondary_vlan_list | add svl | remove svl}
```

- Associates secondary VLANs with the primary VLAN

```
Switch#show vlan private-vlan type
```

- Verifies private VLAN configuration

Configuring Private VLAN Ports


```
Switch(config-if)#switchport mode private-vlan {host |
promiscuous}
```

- Configures an interface as a private VLAN port

```
Switch(config-if)#switchport private-vlan host-association
{primary_vlan_ID secondary_vlan_ID
```

- Associates an isolated or community port with a private VLAN

```
Switch(config-if)#private-vlan mapping primary_vlan_ID
{secondary_vlan_list | add svl | remove svl}
```

- Maps a promiscuous PVLAN port to a private VLAN

```
Switch#show interfaces private-vlan mapping
```

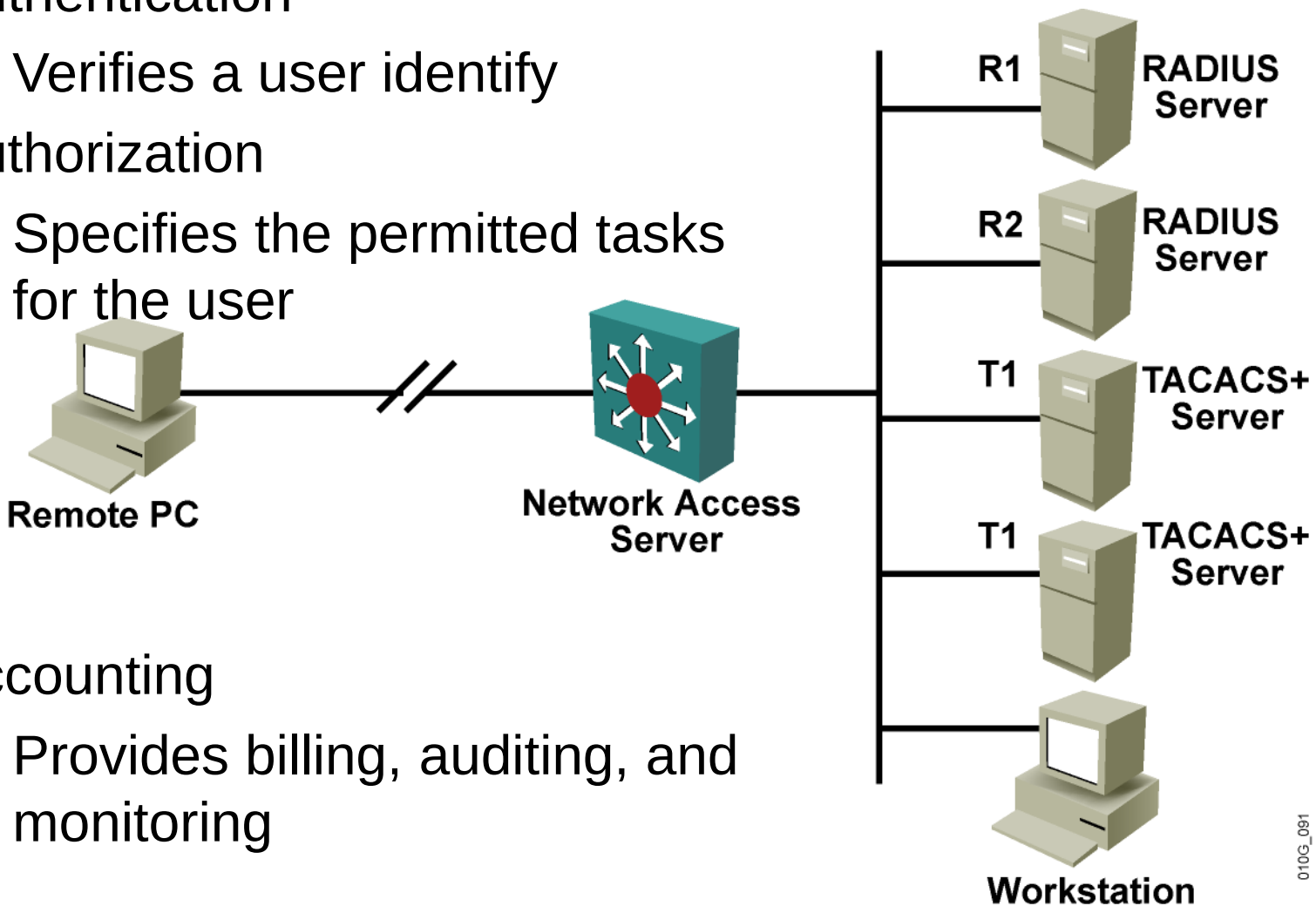
- Verifies private VLAN port configuration

Summary

- VLAN hopping can allow Layer 2 unauthorized access to another VLAN.
- VLAN hopping can be mitigated by:
  - Properly configuring 802.1Q trunks
  - Turning off trunk negotiation
- Access lists can be applied to VLANs to limit Layer 2 access.
- Configuring VLAN ACLs can be accomplished on Cisco Catalyst switches.
- PVLANs are configured to allow traffic flows to be restricted between ports within the same VLAN.
- PVLAN configurations can be applied to provide layer 2 isolation between VLANS.

AAA Network Configuration

- Authentication
  - Verifies a user identify
- Authorization
  - Specifies the permitted tasks for the user
- Accounting
  - Provides billing, auditing, and monitoring



Authentication Methods

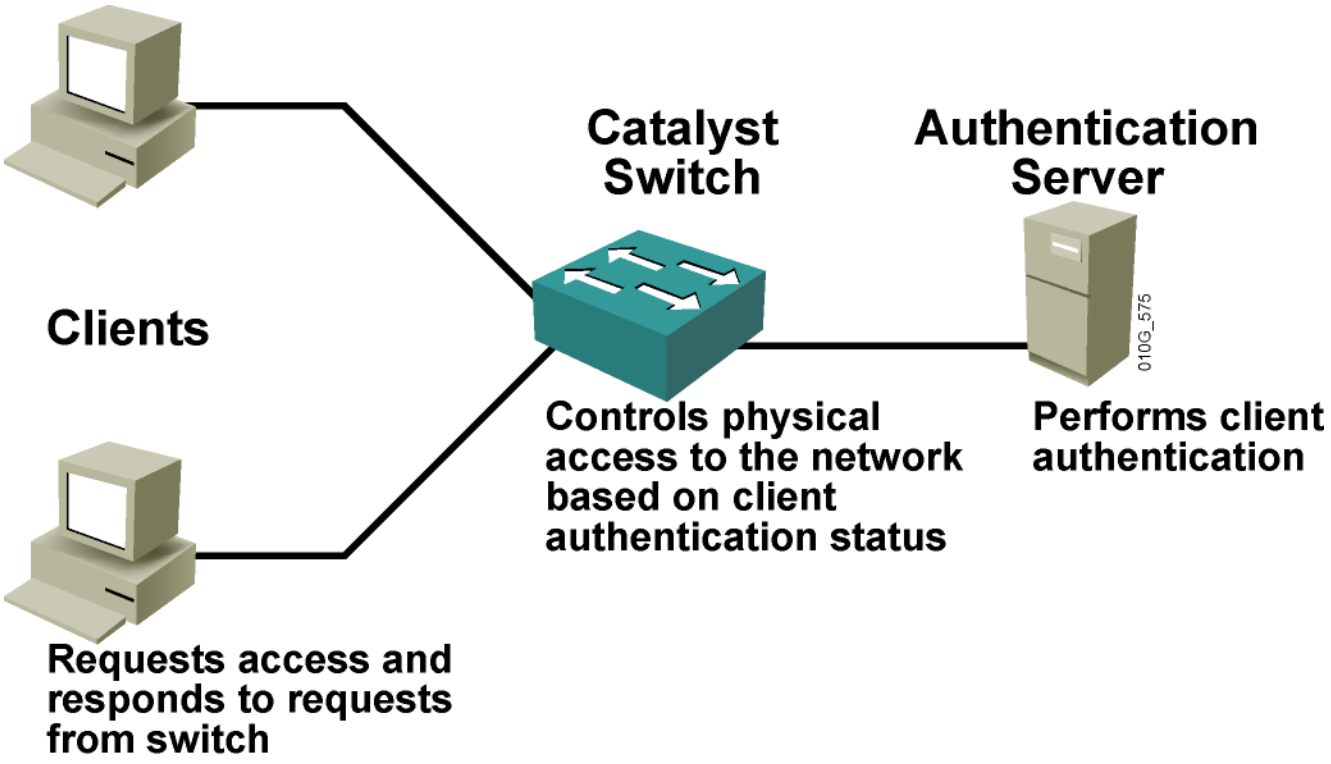

```
Switch(config)#aaa authentication login {default |
list-name} method1 [method2...]
```

- **Creates a local authentication list**

Cisco IOS AAA supports these authentication methods:

- Enable password
- Kerberos 5
- Kerberos 5-Telnet authentication
- Line password
- Local database
- Local database with case sensitivity
- No authentication
- RADIUS
- TACACS+

802.1x Port-Based Authentication



Network access through switch requires authentication.

Configuring 802.1x


```
Switch(config)#aaa new-model
```

- Enables AAA

```
Switch(config)#aaa authentication dot1x {default} method1
[method2…]
```

- Creates an 802.1x port-based authentication method list

```
Switch(config)#dot1x system-auth-control
```

- Globally enables 802.1x port-based authentication

```
Switch(config)#interface type slot/port
```

- Enters interface configuration mode

```
Switch(config-if)#dot1x port-control auto
```

- Enables 802.1x port-based authentication on the interface

Summary

- Layer 2 security measures must be taken as a subset of the overall network security plan.
- Rogue access to the network can undermine the security.
- Switch attacks fall into four main categories.
- MAC flood attacks are launched against Layer 2 access switches and can overflow the CAM table.
- Port security can be configured at Layer 2 to block input from devices.
- Configuring port security on a switch is easy and recommended.
- Sticky MAC addresses allow port security to limit access to a specific, dynamically learned MAC address.
- Multilayer switches should be configured to support security.
- AAA can be used for authentication on a multilayer switch.
- 802.1x port-based authentication can mitigate risk of rogue devices gaining unauthorized access.

Switch Security Issues: Mitigating VLAN, Spoof, and STP Attacks

Part 2: VLANs Attacks

Cabrillo College

CIS 187 Multilayer Switched Networks

CCNP 3 version 4

Rick Graziani

Fall 2006